System Test Report

(TINF18C, SWE I, II Psraxisprojekt 2019/2020)

Projekt: AMLEngine-DLL Interfaces

Kunde : Rentschler & Ewertz
Rotebühlplatz 41
70178 Stuttgart

Lieferant: Team 4 Joshua, Kevin, Krister, Lucas, Markus, Robin
Rotebühlplatz 41
70178 Stuttgart

Version	Datum	Autor	Kommentar
0.1	27.04.2020	Joshua, Kevin, Krister, Lucas, Markus	Dokument erstellt
0.2	01.05.2020	Joshua, Kevin, Krister, Lucas, Markus	Testdaten hinzugefügt
1.0	06.05.2020	Kevin	Kleine Anpassungen vor Veröffentlichung
1.1	11.05.2020	Markus	Final review

1. Scope

The STR (System Test Report) is a document derived from the STP (System Test Plan). It references the tests specified in the STP and documents the results of testing.

2. Definitions

AML AutomationML
AMLX AutomationMLContainer
CLI Command User Interface
DLL Dynamic Linked Library
TC Testcases
TS Testsuite

3. Product names and attributes

The following test objects must be verified:

Ref.-Id.	Product Number	Product Name	Product Description
1	Version 1.0	C++ Wrapper	Compiler settings to allow the usage of the AMLEngine.dll in a C++ project
2	Build 1.0	Console Application	CLI for Validating and (De-)Compressing of AML Files
3	1.0.5	Javascript Wrapper	Wrapper for AmlEngine.dll in Node JS

4. Test Equipment

The needed test equipment for all the tests are specified in the testing setup in the STP.

5. Appendix: Test Cases

5.1. Testsuite <TS-001 C++ Wrapper>

5.1.1. <TC-001-001> (Follow Wrapper instructions)

Testcase ID:	TC-001-001
Testcase Name:	C++ Wrapper
Req.-ID:	UC.001, /LF10/C++ Functions, /LF20/C++ usability
Description:	This test case verifies that the C++ wrapper instructions are understandable and lead to a correct executable, which uses the AMLEngine.DLL.

Test Steps

Step	Action	Expected Result	Actual Result
1	Install Visual Studio Community or Enterprise Microsoft.	The editor can be started.	Editor could be opened.
2	Download the wrapper instructions from the official github repository of this project or open it in the browser	The wrapper documentation can be opened to read.	The wrapper could be opened.
3	Follow the wrapper instructions	A code example using the AMLEngine.dll is ready to compile.	The code example could be created
4	Compile the code with help of the wrapper instructions.	An executable is built by the compiler tools.	The Code example was successfully compiled.
5	Run the executable.	The executable runs.	The executable was successfully started and ran.

Tester:	Krister, Kevin, Markus
Date:	01.05.2020
Testcase Result:	PASS

5.2. Testsuite <TS-002 JS Wrapper>

5.2.1. <TC-002-001> Wrapper import

Testcase ID:	TC-002-001		
Testcase Name:	Wrapper import		
Req.-ID:	UC.002, LF30, LF40		
Description:	Validates that the wrapper package can be downloaded and imported into a node project.		
Test Steps			
Step	Action	Expected Result	Actual Result
1	Make sure to have all the required software installed. (See 7.3)	n/a	All the required software was installed
2	Run the command “*npm i amlenginewrapper --save*”	Success message is displayed and the package is added to the package.json file	The success message was displayed and the amlenginewrapper package was added
3	Invoke the interactive node terminal using the command “*node*”	The console displays the message “Welcome to Node.js”	The message “Welcome to Node.js” was displayed

4	Write the command “*wrapper=require('amlenginewrapper');*”	The package is imported without any error messages	The package was successfully imported
Tester:	Joshua		
Date:	01.05.2020		
Testcase Result:	PASS		

5.2.2. <TC-002-002> Valid call and invalid call handling

Testcase ID:	TC-002-002		
Testcase Name:	Valid call and invalid call handling		
Req.-ID:	UC.002, LF30, LF40		
Description:	Validates that the wrapper package can access all supported functions inside the Adapter. Validates that the wrapper package can also handle invalid function calls.		
Test Steps			
Step	Action	Expected Result	Actual Result
1	Make sure to have all the required software installed. (See 7.3)	n/a	All the required software was installed.
2	Run the command “*npm i amlenginewrapper --save*”	Success message is displayed and the package is added to the package.json file	Success message was displayed and the package was correctly added.
3	Invoke the interactive node	The console	Message was

	terminal using the command “*node*”	displays the message “Welcome to Node.js”	displayed as expected.
4	Write the command “*wrapper=require('amlenginewrapper');*”	The package is imported without any error messages	The package import went as expected without any error messages.
5	Enter the test data specified in the System Test Plan.	The output matches the expected result.	The test data calls worked as expected.
Tester:	Joshua		
Date:	01.05.2020		
Testcase Result:	PASS		

5.2.3. <TC-002-003> Supported functions

Testcase ID:	TC-002-003		
Testcase Name:	Supported functions		
Req.-ID:	UC.002, LF30, LF40		
Description:	Validates that all supported functions can be called using their quick access option		
Test Steps			
Step	Action	Expected Result	Actual Result
1	Make sure to have all the required software installed. (See 7.3)	n/a	All the required software was installed
2	Run the command "*npm install AMLEngineDLLWrapper --save*"	Success message is displayed and the package is added to the package.json file	The success message was displayed and the amlenginewrapper package was added
3	Invoke the interactive node terminal using the command "*node*"	The console displays the message "Welcome to Node.js"	The message "Welcome to Node.js" was displayed
4	Write the command "*wrapper=require('amlenginewrapper');*"	The package is imported without any error messages	The package was successfully imported
5	Enter the test data specified below using the following syntax: wrapper.<function>(<parameters>);	The output matches the expected result.	The test data was entered and the results matched the expected results
Tester:	Joshua		
Date:	01.05.2020		
Testcase Result:	PASS		

5.3. Testsuite <TS-003 Console Application>

5.3.1. <TC-003-001> UI-Test

Testcase ID:	TC-003-001		
Testcase Name:	UI-Test		
Req.-ID:	(Usability)		
Description:	This Test Case verifies the Usability of the Console-Application		
Test Steps			
Step	Action	Expected Result	Actual Result
1	Download and start the Console Application in Windows 10	A Console Application should start and the Main Menu should appear	The Console Application starts and the Main Menu appears.
2	Check if the Main Menu is written correctly and that the UI is user friendly and understandable.	It should be user friendly, understandable and correct.	The Main Menu is user friendly, understandable and has no writing errors.
3	Check if the Options Menu is written correctly and that the UI is user friendly and understandable.	It should be user friendly, understandable and correct.	The Options Menu is user friendly, understandable and has no writing errors.
4	Check if the Validation Menu is written correctly and that the UI is user friendly and understandable. For this a File from Example Files should be verified.	It should be user friendly, understandable and correct.	The Validation Menu is user friendly, understandable and has no writing errors.

5	Check if the DeCompress Menu is written correctly and that the UI is user friendly and understandable. For this a AMLX File from Example Files should be used.	It should be user friendly, understandable and correct.	The Decompress Menu is user friendly, understandable and has no writing errors.
6	Check if the Compress Menu is written correctly and that the UI is user friendly and understandable. For this the Files from Step 5 should be Compressed.	It should be user friendly, understandable and correct.	The Compress Menu is user friendly, understandable and has no writing errors.
Tester:	Markus		
Date:	01.05.2020		
Testcase Result:	PASS		

5.3.2. <TC-003-002> Validation Test

Testcase ID:	TC-003-002		
Testcase Name:	Validation Test		
Req.-ID:	LF70		
Description:	This Test verifies the Validation Functionality of the Console Application		
Test Steps			
Step	Action	Expected Result	Actual Result
1	From the start menu go to the “options” interface. There select the following parameters: 1. “AutoRepair” 2. “PrintAllVal” For best test coverage, try steps 2 and 3 with all 4 combinations. This means use the following settings: • 1– false, 2– false • 1– true, 2– false • 1– false, 2– true • 1– true, 2- true	The output from step 2 and 3 should be adjusted accordingly to the settings made in the options menu.	n/a
2	Validate a correct File using the Validation Menu.	It should validate correctly.	The File has been correctly validated
3	Validate an incorrect File using the Validation Menu. Try both Options, if possible, of Repair the Error and Override the old File. (Depending on the	It should show an error and the option to Repair it. (Depending on the set Options from the Menu)	An Error appears and the Option to repair it.

	set Option from the Menu)		
Tester:	Markus		
Date:	01.05.2020		
Testcase Result:	PASS		

5.3.3. <TC-003-003> (De-)Compression Test

Testcase ID:	TC-003-003		
Testcase Name:	(De-)Compression Test		
Req.-ID:	LF60		
Description:	This Test verifies the Compress and Decompress Functionality of the Console Application.		
Test Steps			
Step	Action	Expected Result	Actual Result
1	Go to the Decompress Menu and Decompress an AMLX File (you can use the File in Example Files but you don't have to).	The File should be Decompressed successfully.	The File was correctly decompressed.

2	Go to the Compress Menu and Compress one File. (You can use the Files from the DeCompression if you want to, but you don't have to)	The File should be Compressed successfully.	The File was correctly compressed.
3	Go to the Compress Menu and Compress two or more Files, but don't use a ClassModel. (You can use the Files from the DeCompression if you want to, but you don't have to)	The Files should be Compressed successfully.	The Files were correctly compressed.
4	Go to the Compress Menu and Compress two or more Files and use a ClassModel. (You can use the Files from the DeCompression if you want to, but you don't have to)	The Files should be Compressed successfully.	The Files were correctly compressed.
Tester:	Markus		
Date:	01.05.2020		
Testcase Result:	PASS		

5.3.4. <TC-003-004> Import Test

Testcase ID:	TC-003-004		
Testcase Name:	Import Test		
Req.-ID:	LF50		
Description:	This Test verifies the Import Functionality per Startparameter of the Console Application.		
Test Steps			
Step	Action	Expected Result	Actual Result
1	The Console Application has the following Parameters: ● path and a valid path after that ● validate -> declares that the File should be validated ● compress -> declares that the File should be compressed This can look like this: “ConsoleApplication.exe --path C:\File.aml -- validate” The First Test Step is that you should Test the Functionality with a valid path and every combination (no parameters, all parameters etc.)	The Console Application should give correct Errors and work correctly.	The Console Application worked correctly with correct Parameters.
2	Test the Functionality with an invalid path and every combination of parameters	The Console Application should give correct Errors.	The Console Application threw correct Errors.
Tester:	Markus		
Date:	01.05.2020		
Testcase Result:	PASS		